SOFTWARE PAY PER USE SYSTEM

Field of the Invention

This invention relates generally to the protection of computer software from illegal copying. More particularly, this invention relates to a pay per use protection technique in which a user must make a telephone call in order to use protected software.

Background of the Invention

Software piracy, the illegal copying of commercial computer programs, is a major problem in the personal computer industry. Software developers currently have three common techniques to prevent illegal copying of their software. The first is the use of hardware "keys". These keys are devices that attach to a computer's printer port. Before starting, the software attempts to query the key. If there is no response, or an incorrect response, the software will not start. The second technique is the use of a key disk. This is like a hardware key, except the key is a floppy disk that must be in the computer's disk drive. A third technique is to sell copy-protected software. In this technique, the developer puts the program on a floppy disk in such a form that a direct copy of the disk will produce an incorrect copy of the software.

All of the above mentioned techniques have problems. They require developers to incorporate piracy protection into their programs. They are brittle, meaning that once a single copy with the protection circumvented exists, the protection is useless. Also, they cause inconvenience to legitimate users of the software. An ideal protection scheme would make it impossible for people to use commercial software illegally, without posing any inconvenience to the developer or the user of the software.

This Sheet Has Been Replaced

REPLACED SHEETS SG
Description : paged ...1... to paged ...4...
Claim(s) : paged ...17... to paged ...22...
Drawing(s) : paged ...4/5... to paged ...5/5...
Abstract : paged ...23... to paged ...-...
DATE OF RECEIPT .....

Summary of the Invention

Generally, the present invention provides a technique which is close to the ideal. It is based upon distributing software in such a way that users must call a telephone number to get the software to work. If the telephone call is to a number which results in a per call service fee, such as a "900" number, then making the call guarantees that the users pay for the use of the software.

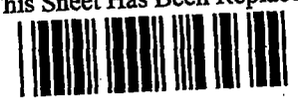
More specifically, an encryption program converts any existing program into a new program called the secured software. The secured software is a program that contains the original software in encoded form. The encryption program, when constructing the secured software, both compresses and encodes the original program. The compression removes most of the redundancy, making code-breaking difficult, and reduces the size of the secured software. The resulting program contains the original software in a completely unreadable form.

The developer may then distribute the secured software freely. To use the software, the user calls a telephone number, from which the user gets a number that the secured software uses to decode the encrypted computer program. This telephone number may be a number that results in a per call service fee, in which case the act of making the call guarantees that the user pays for the use of the software.

Several variations are possible. For example, the secured software may control dialing the telephone number through a modem and running the code transaction automatically. A single call might increment a counter in the software to allow multiple uses. Free telephone service, such as an "800" number, could readily be provided through which potential users can download secured software. Additional services might be readily provided.

To the software developer, the present invention provides an additional distribution source for which there

This Sheet Has Been Replaced



is little cost --the encryption program in its simplest form does not have to be incorporated into the software. To the user, there is no difficulty with backups and no danger of losing a hardware or floppy disk key. All that is needed to run the software is a phone call. The user also has the option of trying out the software before spending a considerable amount to purchase it.

Description of the Drawings

Fig. 1 is a block diagram of an implementation of the system of the present invention.

Fig. 2 is a block diagram illustrating software registration and generation of secured software.

Fig. 3 is a block diagram of a first embodiment of a pay per use software validation technique.

Fig. 4 is a block diagram of a second embodiment of a pay per use software validation technique adding random number generation by the validation system.

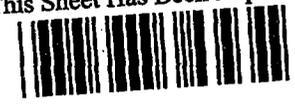
Fig. 5 is a block diagram of a third embodiment of a pay per use software validation technique adding encryption of the generated random numbers.

Detailed Description

This invention is described with reference to various data encoding and decoding techniques. Thus, a brief explanation of basic cryptography follows.

Cryptographic systems generally transform data through the use of two basic elements, a cryptographic algorithm and keys. The cryptographic algorithm contains procedures for encoding and decoding the data. These encoding and decoding procedures are generally identical or may consist of the same steps performed in reverse order. The keys, which are selected by the users, consist of a sequence of numbers or characters, which are used by the cryptographic algorithm to code and decode the data. In the following detailed

This Sheet Has Been Replaced



description we discuss two types of cryptographic systems.

The first type of system is the single key system. In this type of system a single key is used for both data encoding and decoding. Thus, in order to ensure protection, this key must be kept secret. In this description we refer to the Data Encryption Standard (DES) single key technique. This is a data encryption technique which has been accepted as a standard by the National Bureau of Standards, and is well known to one skilled in the art of cryptography.

Another type of system is a public key system. In this type of system, instead of using one key for both data encoding and decoding, two keys are used, one to encode the data and one to decode the data. Generally one key is made public and one key is kept private. If the public key is used to encode the data, then the private key is used to decode the data, and vice versa. An important aspect of this type of system is that it is impossible to deduce the private key from the public key. Public key cryptography is also well known to one skilled in the art of cryptography.

The data encryption and decryption techniques we discuss here are for illustrative purposes. Various other techniques could be substituted for those described herein without departing from the scope and spirit of the invention.

Fig. 1 illustrates one possible implementation of a system according to the present invention. The software validation system 100 comprises a central processing unit 102, a memory unit 110, and a modem 104. The central processing unit 102 is connected to the modem 104 and the memory unit 110. The memory 110 contains software registration program code 111 to implement the functions required during the software registration process, software validation program code 112 to implement the functions required during the software validation process, and a storage area 113 for the storage of software identification

This Sheet Has Been Replaced



information and associated keys. The modem 104 is used to transmit data to and receive data from the software developer system 120 and the software user system 140.

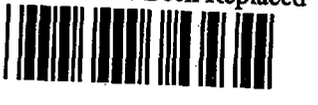
The software developer system 120 comprises a central processing unit 124, a modem 122, and a memory unit 130. The central processing unit 124 is connected to the modem 122 and the memory unit 130. The memory unit 130 contains original software 131, an encryption program 132, and secured software 133 after it is generated. The original software 131 is the software the developer wishes to register and protect. The encryption program 132 is the program which interacts and communicates with the software validation system 100 during software registration and generation of the secured software 133. The secured software 133 contains an encrypted version of the original software 131 along with user validation software. This user validation software is program code that will interact and communicate with the software validation system 100 during software validation. These functions are described below in conjunction with Figs. 2-5.

The software validation system 100 and the software developer system 120 communicate with each other over a communications network 160 through the modems 104 and 122. In one embodiment, the communications network 160 is a public telephone line.

The user system 140 comprises a central processing unit 144, a modem 142, and a memory unit 150. The central processing unit is connected to the modem 142 and the memory unit 150. The memory unit 150 contains a copy of the secured software 151, which comprises both the encrypted original software and the user validation software as discussed above. This secured software 151 is a copy of the secured software 133 which was generated by the software developer system 120.

The validation system 100 and the user system 140

This Sheet Has Been Replaced



communicate with each other over a communications network 170 through the modems 104 and 140. In one embodiment, this communications network 170 is a public telephone line, and the communication is initiated by the user system 140 dialing a telephone number which results in a per call service fee (e.g. a fee set for dialing a "900" number). This call would ensure that the user pays for each use of the software.

The generation of the secured software 133 and the registration of the software with the software validation system is described with reference to Fig. 2. The broken line 202 represents the separation between the software developer system 120 and the software validation system 100. Figure elements shown above line 202 represent functions which are performed by the software validation system 100 by execution of the software registration code 111, and figure elements shown below line 202 represent functions which are performed by the software developer system 120 by execution of the encryption program 132. Data which is sent between the two systems must be transmitted across the communications network 160. The transmission of data over the communications network 160 is represented in Fig. 2 by lines crossing dividing line 202.

The first step 210 is for the software developer to collect software identification information. This information consists of the name of the software, the name of the software developer, the address of the software developer, and any other information which may be desired. This information is transmitted across the network 160 to the software validation system 100. In step 204 the software validation system 100 will store the software identification information in memory 113 and will select the next ID number for the software. This ID number is any unique identifier for the software. The validation system 100 then generates a random public/private key pair and

This Sheet Has Been Replaced

generates a random DES key in step 206.

The generated random private key and the generated random DES key are stored in the software registration system memory 113 along with the software ID number in step 208. Returning now to the software developer system 120, the original software 220 is converted into a packed file in step 218. The method used to pack the original software may be any suitable data compression technique, such as Huffman encoding, which is well known in the art. This compression removes most of the redundancy in the software, making code breaking difficult, and reduces the size of the secured software. The file header from this packed file is then encrypted in step 212 using the DES key generated by the software validation system. Only the file header is encrypted since the packed file cannot be unpacked without the file header. Thus, sufficient protection is ensured by encrypting only the file header. The packed file body and the DES encrypted header which were generated by the software developer system 120, and the public key and the software ID which were generated by the software validation system 100, are then used to build the secured software in step 214. The secured software 216 may then be distributed to users for use in accordance with the invention. In order for a user to use the secured software, it must be converted into an executable module in accordance with the present invention.

A first embodiment of a pay per use validation technique is described in conjunction with Fig. 3. The broken line 302 represents a separation between the software validation system 100 and the user system 140. Figure elements shown above line 302 represent functions performed by the software validation system 100 by execution of the software validation code 112, and figure elements shown below line 302 represent functions performed by the user system 140 by execution of the user validation software

This Sheet Has Been Replaced

portion of the secured software 151. Any data which is passed between the pay per use validation system 100 and the user system 140 is represented by lines crossing dividing line 302 and must be transmitted over the communications network 170. Data is most vulnerable to unauthorized access by an unauthorized user when it is transmitted over the communications network 170.

The first step 316 is to transmit the software ID number to the validation system 100. This is the unique ID number which was assigned to the software during the generation of the secured software (described in conjunction with Fig 2). In step 308 the validation system 100 will use this ID number to credit the software developer's account for the use of the software. As discussed above, in one embodiment of the invention, a user must call a per call service fee telephone number in order to initiate communication over the communications network 170. Thus, the ID number allows the validation system 100 to credit the account of the developer of the software which is being validated. This ID number is also used to look up the private key in step 304 and the DES key in step 306, both of which were generated and stored in the validation system memory 113 during the software registration and generation of the secured software.

The user system generates a random number (R1) in step 320 and encrypts that random number with the public key in step 318. The random number is encrypted so that when it is transmitted to the validation system 100 over the communications network 170, a person attempting to circumvent the protection scheme could not intercept the random number. The validation system 100 will use the private key obtained in step 304 to decrypt the random number generated by the secured software in step 310. The decrypted random number (R1) is then exclusive ORed with the DES key in step 312. This results in a DES key masked by

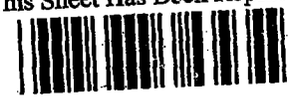
This Sheet Has Been Replaced

the random number generated by the user system. This masked DES key is then encrypted with the private key in step 314 and transmitted to the user system 140 over the communications network 170. The user system will then use the public key to decrypt the masked DES key in step 322. The result is the DES key masked by the random number (R1). This masked DES key is then unmasked by exclusive ORing it with the random number (R1) in step 324. The result is an unmasked, unencrypted DES key. This is the same DES key which was used to encrypt the original software. The DES key is then used to decrypt the encrypted file header in step 326. The result is an unencrypted file header which is then used to unpack the packed file body in step 328. The result is a software executable file which may then be executed 330 on the user system.

As discussed above, the weakest points in this validation process are where data is transmitted across the communications network 170 because it is at these points that an unauthorized user could most easily attempt to intercept and record the data that is being transmitted. One way to circumvent the secured software protection would be to alter the random number generation portion of the user validation code in the secured software 151 so that the same random number is always generated. Then the user calls the telephone number once, giving the fixed random number and records what comes back. Since the program has been altered to always generate this same random number, the information that was recorded is provided to the secured software upon each subsequent execution. One scheme to prevent such unauthorized use is described below in conjunction with a second embodiment of a pay per use validation technique.

This second embodiment is described in conjunction with Fig. 4. In this embodiment, a second random number is generated in the validation system 100. This random number is concatenated with the random number from the user system

This Sheet Has Been Replaced



140, and the combined random number is used to mask and unmask the DES key. This will be clear from the following description of this embodiment in conjunction with Fig. 4. This second embodiment is similar to the first embodiment shown in Fig. 3. Thus, only the differences between the two embodiments will be discussed here. Like numbered elements in Figs. 3 and 4 perform like functions.

In the second embodiment, both the validation system 100 and the user system 140 generate random numbers at steps 320 and 332. In step 334 the validation system 100 concatenates the random number (R1) it generated and the random number (R2) generated by user system. The resulting random number (R1R2) is then exclusive ORed with the DES key in step 312 to produce a masked DES key as discussed in conjunction with Fig. 3. The user system 140, like the validation system 100, concatenates the random numbers R1 and R2 in step 336. The resulting random number (R1R2) is then used to unmask the DES key in step 324 as discussed above in conjunction with Fig. 3. The remainder of the functions of embodiment two are the same as described in conjunction with embodiment one and Fig. 3.

This second embodiment as shown in Fig. 4 adds extra protection to the first embodiment discussed in conjunction with Fig. 3. Since the validation system 100 also generates a random number and then concatenates that random number with the random number generated by the user system 140, the masking of the DES key depends upon both generated random numbers. Thus, even if a user could fix the random numbered generated by the secured software to be the same each time, the random number generated by the validation scheme would be different and therefore, the user could not correctly unmask the DES key.

A third embodiment of a validation technique according to the present invention is discussed in conjunction with Fig. 5. This embodiment adds several features to the

This Sheet Has Been Replaced

embodiments shown in Figs. 3 and 4. The random number (R1) generated in the validation system 100 is encrypted before being transmitted to the user system 140. Also, the random number (R2) generated by the user system 140 is not itself transmitted to the validation system 100. These details will become clear from the following discussion of Fig. 5. The elements of the embodiment shown in Fig. 5 which are the same as those already discussed in conjunction with Figs. 3 and 4 will not be discussed in detail here. Only the additional steps of the embodiment shown in Fig. 5 will be described below. Elements in Fig. 5 with like numbers to elements in Figs. 3 and 4 perform like functions.

Referring now to Fig. 5, the random number (R1) generated by the validation system in step 332 is encrypted with the private key in step 340 before being transmitted to the user system. The encrypted random number is decrypted by the user system in step 346 and is then concatenated in step 336 with the random number (R2) generated by the user system in step 320. The resulting concatenated random number (R1R2) is then encrypted with the public key in step 318 and transmitted to the validation system. The encrypted random number is then decrypted by the validation system in step 310. The validation system, in step 342, then checks to determine whether the random number it generated in step 332 (R1) is the same as the R1 portion of the random number returned by the secured software. If R1 has been modified, it indicates that the random number has been manipulated in some way by the user in an attempt to defeat the protection, and the process is halted. Otherwise, the validation system uses the concatenated random number to mask the DES key in step 312. The system then continues in a manner similar to that explained above in connection with the embodiments of Figs. 3 and 4.

In the embodiment of Fig. 5, the random number (R1) is encrypted before being transmitted from the validation

This Sheet Has Been Replaced

system to the user system. Similarly, the concatenated random number R1R2 is encrypted before being transmitted from the user system to the validation system. Note that the random number (R2) generated by the user system is never itself transmitted to the validation system. These techniques make it more difficult for an unauthorized user to defeat the protection scheme by altering the random number generated by the user system 140.

There are other possible techniques for the prevention of piracy by altering the random number generation. One technique is to not generate or keep a direct, complete copy of the random number in the user system. This approach will make it difficult for a user to determine exactly what the current random number is by directly examining memory. For example, the random number could be provided by adding or exclusive ORing several bytes in memory. The random number is then generated in the user's system only when needed and only one byte (or word) at a time exists in a readable form in the system. Another technique to prevent the altering of the random number generation is to do a cyclical redundancy check of the secured program code to make sure it has not been modified to provide a fixed random number. Another technique is to include as part of the random number some easily verified information, such as the approximate time. Then, when the secured software code uses the random number, it can determine if the time information is correct. If not, the random number may have been altered and the system could abort. These techniques for preventing piracy through the random number generation are given as examples only and are not exhaustive. One skilled in the art could implement these, and other techniques in order to prevent piracy through manipulation of the random numbers generated by the user system 140 and the software validation system 100.

Another possible way to attempt to circumvent the software protection could be to examine memory of the user

This Sheet Has Been Replaced



system after the DES key is received and decoded. This approach might reveal the DES key and could render the security features of the system ineffective. There are several techniques which would prevent this piracy. One technique is to split the program code into several segments each with a separate DES key. The above described decryption techniques could be repeated several times to prevent having all necessary DES keys in memory simultaneously. Another technique would be to store segments of the DES key in several places in memory so as to make finding the entire DES key more difficult. Another technique makes use of the fact that storing segments of the DES key in indirect form in memory will make determining the key difficult. For example, by exclusive ORing sections of the key with other random sections of memory when needed, would prevent an exact copy of any part of the DES key from ever existing in memory. Examination of several locations in memory would be necessary to determine each byte of the DES key. One skilled in the art could implement these and other techniques to prevent piracy by examining memory to determine the DES key.

Another possible piracy technique could be to examine and copy the memory of the user system after the secured software has been decoded. At this point, the original software exits in an executable format in the user system. There are several possible techniques to prevent this type of piracy. For example, an interrupt routine based on a timer interrupt could be added to the secured software. This routine would examine the program counter in the user system central processing unit and would eliminate or overwrite the program from memory once the program counter information indicates that the program is no longer running. Another technique is to arbitrarily complement sectors on the disk where the program is stored to indicate that the program is running and to complement those sectors again

This Sheet Has Been Replaced



when the program is finished in order to restore those sectors. This would make a copy of the program from memory that does not go through the normal startup/finish sequence damaging to the information on the disk. The program state could not be captured at an arbitrary point, saved to disk, and later restored for further executions. The disadvantage is that power outages, resets, or other abnormal terminations would leave the disk in the altered state. Another technique would be to encode the system time into the program's memory image, and check it with the current system time on a frequent basis. The program could be eliminated if the system time has large discontinuities that indicate the program may have been saved and restored for later execution. Yet another technique would be to encode sensitive information about the user into the program's memory image. This could be a credit card number or the user's phone number. This would not only discourage giving away copies of executable memory images illegally saved, but would also allow traceback to the source of the de-secured or altered software. One skilled in the art could implement these and other techniques to prevent piracy by copying an executable copy of the program from memory.

In addition to the above, various other security enhancements are possible. For example, a developer's toolkit could provide methods of charging other than on a per use basis. This might include charging by the number of files created, by the hour or day, etc. Also, it might be possible to allow single calls to more expensive telephone numbers to authorize multiple uses. For example, if each use cost $0.75 on a per use basis, a call to another number might cost $5.00 and authorize 10 uses, while another number might cost $50.00 and authorize 500 uses, giving the user a substantial volume discount. These "developer's toolkit" derived versions of the secured software would have to be able to save themselves on disk in a semi-secured form with

This Sheet Has Been Replaced



a software counter that keeps track of the number of executions/operations left. It would also be considerably easier to make unauthorized copies and illegally distribute copies with the "counters" set to high values. Embedding the user's verified credit card number and phone number would probably prevent most people from distributing illegal copies. Another way to trace illegal copies of semi-secured discount volume usage software would be to randomly, for example once every 5 - 10 executions, erase a byte of the program's image from memory and disk, and then require the user to call a telephone number. The user would then provide the number displayed on the screen, for example the program ID and the erased address, and the telephone response system would give back what was missing. In this way a record of telephone numbers is generated of users with semi-secured volume usage software. If a comparison of this number list generated large call volumes from numbers that were not on the per call service fee number list, then those numbers may have illegal copies of unsecured software. With a developer's toolkit, random combinations of security methods might be applied to a particular piece of software so that the methods for securing different pieces of software are different. Developing a method to break one software package would not be applicable to later versions of the same or different software packages. These and other security techniques to prevent the unauthorized decryption of the secured software could be readily implemented by those skilled in the art.

In each of the embodiments described above, there are several possible variations for the transmission of data between the validation system 100 and the user system 140. In the simplest, a user may call a telephone number and speak to a person who has access to the validation system. The two human operators could communicate the information orally. The software user would then provide the required

This Sheet Has Been Replaced



information to the user system. Another variation would be for the user to call a telephone number which connects the user to an automated response system connected to the validation system. In this case, the user could use a telephone keypad to pass information to the validation system, and the validation system could pass information back to the user by voice synthesis or recording. A further variation would be to transmit data between the systems via modem. In this way, the user system would communicate directly with the validation system via electronic communications. Thus, the validation of the secured software would be almost transparent to the user.

It is to be understood that the embodiments and variations shown and described herein are illustrative of the principles of this invention only and that various modifications may be implemented by those skilled in the art without departing from the scope and spirit of the invention.

This Sheet Has Been Replaced

Claims:

1. A system for the validation and use of secured computer software comprising:

a software validation system;

a software user system comprising secured software;

a communications network connected to the software validation system and the software user system for the transmission of data between the systems;

said software validation system comprising:

means for storing at least one decryption key,

means for receiving from said user system over said communications network an identification of said secured software,

means for recording the receipt of said identification of secured software, and

means for transmitting to said user system over said communications network a decryption key chosen from said at least one stored decryption key, wherein said chosen decryption key is capable of decrypting said identified secured software;

said software user system further comprising:

means for transmitting said secured software identification to the software validation system over said communications network,

means for receiving said chosen decryption key capable of decrypting said identified secured software from said software validation system over said communications network, and



This Sheet Has Been Replaced



means for decrypting said secured software using said chosen decryption key.

2. A software validation system comprising:

means for storing at least one decryption key;

means for receiving from a communications network an identification of encrypted software;

means for recording the receipt of said identification; and

means for transmitting to the communications network a decryption key chosen from said at least one stored decryption key, wherein said chosen decryption key is capable of decrypting said identified encrypted software, and

wherein a fee is charged for the establishment of communication over the communications network.

3. The software validation system of claim 2 further comprising:

means for masking said decryption key with a random number prior to transmitting it to the communications network.

4. The software validation system of claim 3 further comprising means for receiving said random number from the communications network.

5. The software validation system of claim 2 further comprising:

means for generating a first random number;

CLMS

This Sheet Has Been Replaced

means for receiving a second random number from the communications network;

means for producing a third random number by concatenating said first random number and said second random number; and

means for masking said decryption key with said third random number prior to transmitting it to the communications network.

6. The software validation system of claims 3 or 5 further comprising means for encrypting said masked decryption key prior to transmitting it to the communications network.

7. The software validation system of claim 6 wherein said means for encrypting is by use of a public key encryption algorithm.

8. The software validation system of claim 2 further comprising:

means for generating a first random number;

means for encrypting said first random number;

means for transmitting said encrypted first random number to the communications network;

means for receiving an encrypted second random number from the communications network, wherein said second random number comprises a first section and a second section;

means for decrypting said second random number; and

means for masking said decryption key with said second random number prior to transmitting it to the communications network.



This Sheet Has Been Replaced



9. The software validation system of claim 8 further comprising:

means for determining whether said first or second section of said second random number is equal to said first random number; and

means for masking said decryption key with said second random number prior to transmitting only if said first or second section of said second random number is equal to said first random number.

10. A software user computer system for the execution of secured software, the system comprising:

encrypted computer program code;

means for transmitting an identification of the encrypted computer program code to a communications network;

means for receiving a decryption key from the communications network, said decryption key capable of decrypting said encrypted computer program code; and

means for decrypting said encrypted computer program code with said decryption key;

wherein a fee is charged for the initiation of the communication over the communications network.

11. The software user computer system of claim 10 wherein said received decryption key has been masked prior to receipt, the system further comprising:

* C L M S *

This Sheet Has Been Replaced

means for generating a random number;

means for transmitting said random number to the communications network; and

means for unmasking said masked decryption key with said random number.

12. The software user computer system of claim 10 wherein said received decryption key has been masked and encrypted prior to receipt, the system further comprising:

means for decrypting said encrypted masked decryption key;

means for generating a random number;

means for transmitting said random number to the communications network; and

means for unmasking said encrypted masked decryption key with said random number.

13. The software user computer system of claim 10 further comprising means for preventing the unauthorized decryption of said encrypted computer program code if communication over the communications network has not been initiated by a telephone call which results in a per call service fee.

14. A software user computer system for the execution of secured software, the system comprising:

encrypted computer program code;

* C L M S *

This Sheet Has Been Replaced

means for transmitting a software identification identifying the encrypted computer program code to a communications network;

means for generating a first random number;

means for receiving a second random number;

means for combining said first and second random number to produce a third random number;

means for receiving a masked decryption key capable of decrypting said encrypted computer program code;

means for unmasking said masked decryption key with said third random number; and

means for decrypting said encrypted computer program code with said decryption key,

wherein a fee is charged for the initiation of the communication over the communications network.

15. The system of claims 1, 2, 10 or 14 wherein communication over the communications network is initiated by a telephone call which results in a per call service fee.

16. The software user computer system of claim 14 wherein said masked decryption key is encrypted prior to receipt, said system further comprising means for decrypting said encrypted masked key prior to unmasking said key.

17. The software user computer system of claim 14 wherein said second random number is encrypted prior to receipt and said

* C L M S *

This Sheet Has Been Replaced

masked decryption key is encrypted prior to receipt, said system further comprising:

means for decrypting said encrypted masked key prior to unmasking said key; and

means for decrypting said encrypted second random number prior to combining said first random number and second random number.

18. A computer system for the registration of software and the generation of secured software comprising:

means for transmitting software identification information to a communications network;

means for receiving at least one key and a unique software identification from the communications network; and

means for encrypting an executable software file using said at least one key to create a secured software module,

wherein said executable software file is capable of being executed by a user only after receipt of said at least one key.

19. The computer system of claim 18 wherein said secured software module comprises the encrypted executable software file and computer program code, said computer program code comprising said means for decrypting the executable software file using said at least one key, and

CLMS

This Sheet Has Been Replaced

wherein said at least one key is received by a user only after said user places a telephone call which results in a per call service fee.

20. A software validation system for the registration of protected software, the system comprising:

means for receiving software identification information from a communications network;

means for generating a unique software identification code for said received software identification;

means for generating at least one cryptographic key;

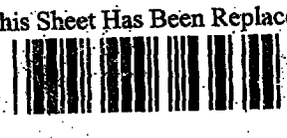
means for transmitting said unique software identification code and at least one cryptographic key to the communications network; and

means for storing said unique software identification and at least one cryptographic key,

wherein said at least one cryptographic key will be used to generate a secured software module, said secured software module being executable by a user computer system only after receipt of said at least one cryptographic key over a communications network, wherein said receipt of said at least one cryptographic key is initiated by a telephone call which will result in a per call service fee.



This Sheet Has Been Replaced



21. A method for the validation and use of encrypted secured software comprising the steps of:

a user of secured software initiating a telephone call which will result in a per call service fee, said telephone call establishing communication over a communications network between said user and a software validation system;

said user of secured software transmitting secured software identification to the software validation system over the communications network;

said software validation system transmitting to said user over the communications network a decryption key capable of decrypting said secured software; and

decrypting said secured software with said decryption key.

22. The method of claim 21 further comprising the steps of:

generating a random number in the user computer system;

transmitting said random number to the software validation system;

masking said decryption key with a random number in said software validation system prior to transmitting the decryption key to the user; and

unmasking said decryption key with said random number in the user's computer system.

23. A method for validating secured computer software comprising the steps of:

* C L M S *

This Sheet Has Been Replaced

storing at least one decryption key;

receiving from a communications network an identification of encrypted software;

recording the receipt of said identification; and

transmitting to the communications network a decryption key chosen from said at least one stored decryption key, wherein said chosen decryption key is capable of decrypting said identified encrypted software,

wherein a fee is charged for the establishment of communication over the communications network.

24. The method of claim 23 wherein said establishment of communication over the communications network is established by placing a telephone call which results in a per call service fee.

25. The method of claim 23 further comprising the step of masking said decryption key with a random number prior to transmitting it to the communications network.

26. The method of claim 25 further comprising the step of receiving said random number from the communications network.

27. The method of claim 23 further comprising the steps of:

generating a first random number;

receiving a second random number from the communications network;

CLMS

This Sheet Has Been Replaced

producing a third random number by concatenating said first random number and said second random number; and

masking said decryption key with said third random number prior to transmitting it to the communications network.

28. The method of claims 25 or 27 further comprising the step of encrypting said masked decryption key prior to transmitting it to the communications network.

29. The method of claim 28 wherein said step of encrypting is by use of a public key encryption algorithm.

30. The method of claim 23 further comprising the steps of:

generating a first random number;

encrypting said first random number;

transmitting said encrypted first random number to the communications network;

receiving an encrypted second random number from the communications network, wherein said second random number comprises a first section and a second section;

decrypting said second random number; and

masking said decryption key with said second random number prior to transmitting it to the communications network.

CLMS

This Sheet Has Been Replaced

31. The method of claim 30 further comprising the steps of:

determining whether said first or second section of said second random number is equal to said first random number; and

masking said decryption key with said second random number prior to transmitting only if said first or second section of said second random number is equal to said first random number.

32. A method for the execution of encrypted secured software comprising the steps of:

initiating communication over a communications network which results in a service fee;

transmitting an identification of the secured software to the communications network;

receiving a decryption key from the communications network, said decryption key capable of decrypting said secured software; and

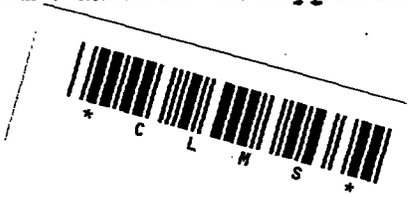
decrypting said secured software with said decryption key.

33. The method of claim 32 wherein said received decryption key has been masked prior to receipt, the method further comprising the steps of:

generating a random number;

transmitting said random number to the communications network; and

unmasking said masked decryption key with said random number.



This Sheet Has Been Replaced



34. The method of claim 32 wherein said received decryption key has been masked and encrypted prior to receipt, the method further comprising the steps of:

decrypting said encrypted masked decryption key;

generating a random number;

transmitting said random number to the communications network; and

unmasking said encrypted masked decryption key with said random number.

35. The method of claim 32 further comprising the step of:

preventing the unauthorized decryption of said encrypted computer program code if communication over the communications network has not been initiated by a telephone call resulting in a per call service fee.

36. A method for the execution of encrypted secured software comprising the steps of:

initiating communication over a communications network which results in a service fee;

transmitting a software identification identifying the secured software to the communications network;

generating a first random number;

receiving a second random number;

* C L M S *

This Sheet Has Been Replaced

combining said first and second random number to produce a third random number;

receiving a masked decryption key capable of decrypting said secured software;

unmasking said masked decryption key with said third random number;

decrypting said secured software with said decryption key.

37. The method of claims 32 or 36 wherein said step of initiating communication over a communications network comprises the step of placing a telephone call resulting in a per call service fee.

38. The method of claim 36 wherein said masked decryption key is encrypted prior to receipt, said method further comprising the step of decrypting said encrypted masked key prior to unmasking said key.

39. The method of claim 36 wherein said second random number is encrypted prior to receipt and said masked decryption key is encrypted prior to receipt, said method further comprising the steps of:

decrypting said encrypted masked key prior to unmasking said key; and

decrypting said encrypted second random number prior to combining said first random number and second random number.

* C L M S *

This Sheet Has Been Replaced

40. A method for the registration of software and the generation of secured software comprising the steps:

transmitting software identification information to a communications network;

receiving at least one key and a unique software identification from the communications network; and

encrypting an executable software file using said at least one key to create a secured software module,

wherein said executable software file is capable of being executed by a user only after receipt of said at least one key.

41. The method of claim 40 wherein said at least one key is received by a user only after said user initiates a telephone call to a telephone number which results in a per call service fee.

42. A software validation method for the registration of protected software comprising the steps:

receiving software identification information from a communications network;

generating a unique software identification code for said received software identification;

generating at least one cryptographic key;

transmitting said unique software identification code and at least one cryptographic key to the communications network; and

* C L M S *

This Sheet Has Been Replaced

storing said unique software identification and at least one cryptographic key,

wherein said at least one cryptographic key will be used to generate a secured software module, said secured software module being executable by a user computer system only after receipt of said at least one cryptographic key over a communications network, wherein said receipt of said at least one cryptographic key is initiated by a telephone call which will result in a per call service fee.

* C L M S *

This Sheet Has Been Replaced